实数编码量子共生演算法及其在云任务调度中的应用 *

李昆仑，关立伟

(河北大学 电子信息工程学院，河北 保定 071000)

摘　要：针对共生演算法收敛慢和易陷入局部最优的问题，结合量子遗传算法理论，提出一种实数编码的量子共生演算法(real-coded quantum symbiotic organisms search，RQSOS)。首先依据三角模糊数提出差异度概念，并依此构造一个以自变量向量的分量和一对概率幅为等位基因的三倍染色体，使一条染色体携带更多信息并增强解的多样性；然后提出一种基于阿基米德螺旋线的探索学习模式，加强对解空间的探索精度；最后使用共生演算法更新差异度值并依据差异度值对种群进行学习和变异操作，促使整个种群快速向最优方向进化且减小了陷入局部最优的概率。利用数值优化问题和云任务调度问题对算法进行验证，仿真结果表明，RQSOS 算法在收敛速度和寻优能力上均有明显提升，是一种可行有效的算法。

关键词：量子遗传算法；共生演算法；差异度；数值优化；任务调度

中图分类号：TP393

Real-coded quantum SOS algorithm and its application in cloud task scheduling

Li Kunlun, Guan Liwei

(*Electronic Information Engineering College Hebei University*, *Baoding* 071000, *Chinaa*)

Abstract: In order to solve the problem that symbiotic organisms search algorithm converge slowly and easy to fall into the local optimum, combining quantum genetic algorithm theory, this paper proposed a real-coded quantum symbiotic organisms search algorithm(RQSOS) . First, this paper presented the concept of the difference degree based on the principle of triangular fuzzy number, and constructed a variable component vector and a pair of probability amplitude of a allele in a chromosome that can carries more information and enhance the diversity of the solutions. Then the mode of rotary learning based on the Archimedes spiral was proposed, which strengthen the exploration ability of the solution space. Finally it is updated the difference degree based on SOS, and the population learning and mutation operations are carried out based on the value of the difference degree which can make the whole population evolution rapidly towards the optimal direction and reduce the probability of falling into local optimum. It is verified by numerical optimization and cloud task scheduling problem, and the simulation results show that the RQSOS algorithm can significantly improve the convergence speed and optimization ability, which is a feasible and effective algorithm.

Key Words: genetic quantum algorithm; symbiotic organisms search; difference degree; numerical optimization; task scheduling

0　引言

量子遗传算法(genetic quantum algorithm,QGA)[1]是Narayanan和Moore于1996年提出的智能优化算法，首次将量子计算相关理论与遗传算法结合，提升了算法性能。随后Han等人[2]将其扩展为量子进化算法(quantum-inspired evolutionary algorithm, QEA)，促进了量子计算理论与智能算法的结合，逐渐形成一个新的研究方向---量子智能算法。到目前为止，具有代表性的量子智能算法有量子进化规划(quantum evolutionary programming, QEP)、量子粒子群算法(quantum particle swarm optimization, QPSO)和量子蚁群算法(quantum ant colony algorithm, QACO)等[3-5]。

量子智能算法能为排序问题、背包问题、路径优化问题和任务调度等问题找到更优解决方案[6-10]。但近些年多位学者证明并非所有量子智能算法都适用于解决连续优化问题和多参数优化问题，且对于高精度高维度计算问题，通过对量子位进行观测而得到的二进制编码方式很容易引起染色体长度灾难，导致求解的精度降低，算法进化效率下降[11-13]。为克服这些缺点，

基金项目：国家自然科学基金资助项目（61672205）

作者简介：李昆仑（1962-），男，河北保定人，教授，博士，主要研究方向为模式识别、图像处理、计算机网络、智能信息处理等；关立伟（1990-），男（满族），河北承德人，硕士研究生，主要研究方向为模式识别、云计算.

以实数编码为主要思想的量子智能算法受到学者们的推崇。基于实数编码量子智能算法使用量子比特编码染色体，利用量子门更新种群，其通过构造量子位到实数部的映射关系，克服了传统量子智能算法解码复杂度高、测量操作易产生退化和“维数灾”等缺点。但量子位到实数的映射关系一般为缩放体制，解码精度不能得到保证，故这些算法依然存在求解精度低、收敛慢和易陷入局部最优的问题。

为了克服上述问题，结合量子遗传算法和共生演算法，提出一种实数编码的量子共生演算法，利用数值优化问题和云任务调度问题对算法性能进行验证。实验结果表明，该方案在提升共生演算法性能的同时，成功避免了上述缺陷。

1 经典量子遗传算法和共生演算法

1.1 经典量子遗传算法

量子遗传算法染色体用量子位表示，一个量子位由$[\alpha,\beta]^T$组成，α,β代表状态$|0\rangle$和$|1\rangle$的概率幅，满足归一化条件$|\alpha|^2+|\beta|^2=1$。一个量子染色体可表征解空间中任意解的叠加态，具体表示如下：

$$\begin{bmatrix}\alpha_1 & \cdots & \alpha_n\\ \beta_1 & \cdots & \beta_n\end{bmatrix} \tag{1}$$

量子遗传算法依据量子门进行演化，促使量子种群趋向最优解。常用的量子门有旋转门、异或门和 Hadamard 门等，应用较广的是如下的量子旋转门。

$$\begin{bmatrix}\alpha_i'\\ \beta_i'\end{bmatrix}=\begin{bmatrix}\cos\theta_i & -\sin\theta_i\\ \sin\theta_i & \cos\theta_i\end{bmatrix}\cdot\begin{bmatrix}\alpha_i\\ \beta_i\end{bmatrix} \tag{2}$$

其中：$[\alpha_i,\beta_i]^T$为第 i 个量子位，$[\alpha'_i,\beta'_i]^T$为旋转之后量子位，θ为旋转角，根据设定调整策略确定其大小。旋转门可以实现状态间的转换，具有较高的并行性，保证算法的收敛。

1.2 共生演算法

共生演算法（symbiotic organisms search, SOS）[14]由 Cheng 等人在 2014 年提出，包含如下三个过程：

a)共生过程。

$$X_{inew}=X_i+rand(0,1)*(X_{best}-Mutual_Vector*BF_1) \tag{3}$$

$$X_{jnew}=X_j+rand(0,1)*(X_{best}-Mutual_Vector*BF_2) \tag{4}$$

$$Mutual_Vector=(X_i+X_j)/2 \tag{5}$$

其中:X_i代表第 i 个位置，X_j是随机与X_i相作用的位置。BF_1和BF_2代表受益水平取值为 1 或 2。

b)共栖过程。

$$X_{inew}=X_i+rand(-1,1)*(X_{best}-X_j) \tag{6}$$

其中:$rand(-1,1)$为[-1,1]的随机数，$(X_{best}-X_j)$反映一种受益关系，由X_j提供一些受益点来提升X_i的存活率。

c)寄生过程。

$$X_{inew}=X_p \tag{7}$$

寄生个体通过随机修改不同维数中的数值产生，该过程能加强种群多样性，防止陷入局部最优。

2 实数编码量子共生演算法

实数编码量子共生演算法是一种融合量子遗传算法思想和共生演算法优点的智能算法。该算法为解决共生演算法前期进化速度慢和后期种群多样性丢失的问题，首先改进量子比特编码形式，构造一个具有关联特性的三倍染色体，然后通过监视染色体信息确定种群进化速度和多样性情况，并依据监视结果适时加强种群进化速度并补充种群多样性，最后利用共生演算法更新染色体，实现问题的优化求解。

2.1 染色体编码

本文提出一种具有关联特性的编码形式，表示为

$$\begin{bmatrix}x_1 & \cdots & x_n\\ \cos(\vartheta_1) & \cdots & \cos\vartheta_n\\ \sin(\vartheta_1) & \cdots & \sin(\vartheta_n)\end{bmatrix} \tag{8}$$

其中：n 为求解问题的维数，x_i 为自变量向量 x 的分量，$\vartheta_i=0.5*\theta_i*\pi$，$\theta$为差异度值，由三角模糊数产生。由$\cos(\vartheta)$控制变异信息，$\sin(\vartheta)$控制趋向信息。

2.1.1 三角模糊数

依据集对理论[15]，三角模糊数特征可由二元联系数$\bar{x}+si$表示，其中$\bar{x}$为均值，s为方差，计算方法如下：

$$\bar{x}=\frac{1}{3}\left[x^l+x^m+x^n\right] \tag{9}$$

$$s=\frac{1}{2}\sqrt{(x^l-\bar{x})^2+(x^m-\bar{x})^2+(x^n-\bar{x})^2} \tag{10}$$

其中：x^l为三角模糊数的下确界，x^m为三角模糊数的最可能值，x^n为三角模糊数的上确界。$\bar{x}$和 s 是对三角模糊数 相对确定与不确定性的描述，在 D-u 空间可表示为如图 1 所示。

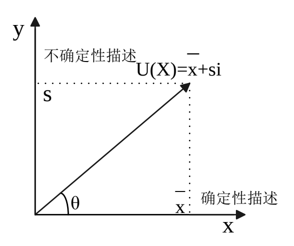


图 1 三角模糊数在 D-U 空间的解释

其中θ表示相互作用的方向，称为特征参数$U(x)$的幅角，计算方式为：$\theta=\arctan s/\bar{x}$。

2.1.2 种群差异度

由 2.1.1 节分析可知，θ值越小($\theta\in[0,\pi/2]$)，确定性越大，波动性越小，因此，种群差异度由下述方式得出：

$$\bar{x}'=\frac{1}{3}\left[x^j_{\min}+x^j_{best}+x^j_{\max}\right] \tag{11}$$

$$s'=\frac{1}{2}\sqrt{(x^j_{\min}-\bar{x})^2+(x^m-\bar{x})^2+(x^j_{\max}-\bar{x})^2} \tag{12}$$

其中：$\bar{x}'$代表最优信息的均值，本文称其为最优均值；$x^j_{\min}$为当前维数最小值，x^j_{best}代表当前维数最优值，$x^j_{\max}$代表当前维数

最大值；计算 s' 时，x^m 为当前位置值，以此计算确定性和波动性关系。故差异度值定义为 $\theta=\arctan s'/\bar{x}'$。

以下举例说明染色体实数部与差异度间关系：

某一维多样性较弱，设 3 为当前最优值，如图 2 左所示。

1	…	3
2	…	3
3	…	2
4	…	3
5	…	3
前期位置i		后期位置i

图 2　染色体实例

通过计算得到：s'=[1.7321 1.5000 1.4142 1.5000 1.7321]'; θ=[0.5236 0.4636 0.4405 0.4636 0.5236]'。可知，当前值与最优均值越接近 θ 角度值越小，相反则越大，但 θ 依然较大。

种群后期多样性较小，如图 2 右所示，通过计算得 s'=[0.4082 0.4082 0.5000 0.4082 0.4082]';θ=[0.1519 0.1519 0.1853 0.1519 0.1519]'。可知，θ 依然满足与最优均值越接近值越小的关系，算法后期多样性差，故差异度值更小。图 3 为 SOS 算法优化 Rosenbrock 函数的平均差异度值变化曲线。

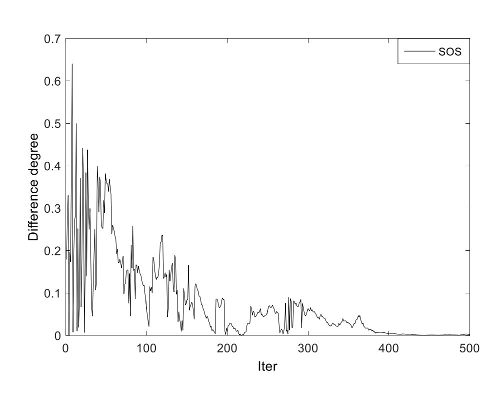


图 3　SOS 算法 θ 值随迭代次数变化曲线

由图 3 可知，种群后期多样性已经严重丢失，算法性能下降。因此，通过对 θ 值的监测，实时了解种群多样性，当种群多样性过大时，应采取措施增加个体优秀基因，可加快算法收敛速度。在种群多样性丢失时及时采取措施可减小算法陷入局部最优的概率，加强算法寻优能力。

2.2　趋向学习和交叉学习

2.2.1 基于阿基米德螺旋线的探索学习模式

文献[16]和[17]证明了反向学习和旋转学习的有效性。但旋转角度的设定影响着算法探索精度，该模式依然存在探索粗糙的问题。为进一步加强算法趋向最优解和探索能力，提出一种基于阿基米德螺旋线的探索学习模式。

如图 4 所示，旋转模式分为两种：a)并行旋转模式，以外围圆半径为半径绕圆心旋转，半径与圆及阿基米德线相交点横坐标即为并行学习点；b)求精旋转模式，以外圆半径为半径绕圆心旋转，产生两个学习点，一个由阿基米德线内部最小圆上点由内向外移动，第二个点依然在外圆上移动，这样一个在外部学习，一个由圆心向外学习，可加快学习速率。具体过程如下：

以 C 为中心(C=(a+b)/2，a,b 为解空间上下界)，(b-a)/2 为半径作外围圆并以式(13)在设定空间内散发阿基米德螺旋线，半径变化范围为 0~(b-a)/2。

$$f(r)=(a',b',\phi), a'\neq 0 \tag{13}$$

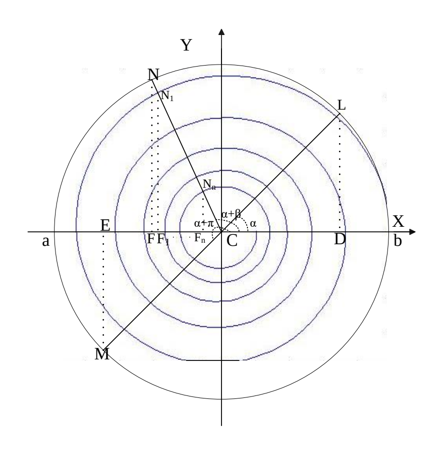


图 4　基于阿基米德螺旋线的探索学习模式

$f(r)$代表螺旋线的产生方式，其极坐标形式为 $r=a'\phi+b'$。学习过程为对当前点进行一定角度 β 的旋转，旋转后位置则为学习后的点，其横坐标为

$$F_i=\left(\frac{a+b}{2}\right)+f(r_i)\cdot\cos(\alpha+\beta) \tag{14}$$

其中:β 为旋转角, α=arccos((2*z-a-b)/(b-a))，$f(r_i)$为旋转位置半径；外围圆上点对应横坐标为

$$F=\left(\frac{a+b}{2}\right)+\left(\frac{a-b}{2}\right)\cdot\cos(\alpha+\beta) \tag{15}$$

基于阿基米德线的两种学习模式采取不同的方式求得学习点，可加强算法的学习精度，提高搜索效率。

2.2.2 趋向学习

由 2.1 节可知，θ 角取值较大(sin(ϑ)>P_i)，种群多样性较好，但与当前最优均值的综合距离相差较大，代表最优信息较少。为加强算法的收敛能力，加快算法向最优方向逼近，本节引入趋向学习操作，使个体在一个扇形空间内趋向最优个体，最大趋向半径为 R'，如图 5 所示，β_{best} 为当前最优位置在外围圆半径与 x 轴夹角，β_i 为当前位置夹角。

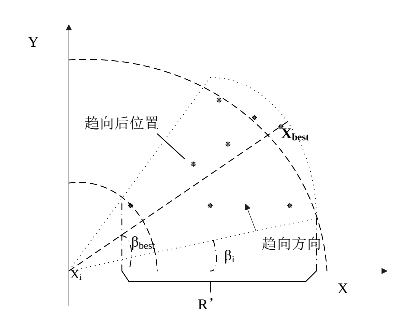


图 5　扇形区域内的趋向操作

扇形空间范围和趋向个体数量可根据实际情况调整，趋向方式可由式(16)~(18)表示。

$$\Delta\theta=-\mathrm{sgn}(A)\cdot\Delta\theta_0\cdot\exp\left(-\frac{\left|\nabla f\left(x_i^j\right)\right|-\nabla f_{j\min}}{\nabla f_{j\max}-\nabla f_{j\min}}\right) \tag{16}$$

$$\nabla f_{j\max}=\max\left\{\left|\frac{\partial f(x_1)}{\partial x_1^j}\right|,\cdots,\left|\frac{\partial f(x_m)}{\partial x_m^j}\right|\right\} \tag{17}$$

$$\nabla f_{j\min}=\min\left\{\left|\frac{\partial f(x_1)}{\partial x_1^j}\right|,\cdots,\left|\frac{\partial f(x_m)}{\partial x_m^j}\right|\right\} \tag{18}$$

其中：$\nabla f\left(x_i^j\right)$为评价函数 $f(x)$在 x_i^j 处的梯度，$\Delta\theta_0$ 为初始旋转角，具体大小视问题确定，-sign(A)控制旋转角度方向，其中 $A=\begin{vmatrix}\cos(\beta_{best}) & \cos(\beta_i)\\ \sin(\beta_{best}) & \sin(\beta_i)\end{vmatrix}$。设定旋转角度为 $\beta_{new}=f(\lambda,\beta_i+\Delta\beta)$，其中 λ 为加速系数，用以确定角度旋转圈数，取值为 $\lambda=d*2\pi$，d 为$[0,q]$之间的随机整数，q 为规定旋转最大圈数。采用模式 2 进行旋转学习，取 n_1 个趋向个体，若趋向个体为有效个体则接受新个体，否则淘汰。

2.2.3 交叉学习

为进一步加强算法性能，提出一种交叉学习模式。具体为：随机选取 M 个待交叉点，采用式(19)决定是否向当前最优个体进行学习，学习有效则接受新个体。

$$x_{inew}^j=\begin{cases}x_{besti}^j, if\ (\sin(\vartheta))^2>P_c\\ x_i^j, \quad else\end{cases} \tag{19}$$

其中：$\vartheta=0.5*\theta*\pi$，上式可解释为：θ 值较大，说明当前维度信息离当前最优均值较远，应大概率接受当前最优第 j 维的值，相反则小概率接受当前最优值。依据差异度值决定是否学习，有助于种群学习的成功率，加快种群进化速度。

2.3 变异

学习操作引导整个种群向当前最优位置进化，由于算法的贪婪选择特性，后期种群多样性可能丢失，算法易陷入局部最优解。基于此，本节提出一种变异操作。

当 θ 值较小，认定种群与当前最优具有极强的相似性，多样性丢失，在此时进入变异操作。依据阿基米德探索模式，利用如下公式产生变异信息：

$$M_j=\begin{cases}x_j+\mathrm{sgn}(\bullet)\cdot f(r) & ,if\ (\cos(\vartheta))^2>\eta\\ x_j \quad , & else\end{cases} \tag{20}$$

$$\mathrm{sgn}(\bullet)=\begin{cases}-1, if\ \mathrm{mod}(\beta_m,2\cdot\pi)\in[0.5,1.5]\\ 1, \quad else\end{cases} \tag{21}$$

其中：sgn(•)为变异方向，$r=\alpha*\beta_m$, $\beta_m=rand*\pi*\mu$，α 为精度控制变量，μ 为$[1,q]$的随机整数，η 为变异阈值。

利用模式 1 在规定空间随机勘测出 n_2 个变异个体并随机替换种群中 n_2 个个体（除最优个体外）。通过变异操作针对性的补充种群多样性，可使算法不易陷入局部最优解。

2.4 算法流程

本文依据 SOS 算法更新实数部进而更新差异度 θ 值，然后依据差异度值来控制该维信息在一定范围内的学习与变异的情况，具体过程如下：

a)初始化实数部种群和最大旋转圈数 q，旋转精度 a，趋向个体数量 n_1，变异个体数量 n_2 和变异率 η 等。

b)通过模糊三角函数形式，计算差异度值并产生三倍染色体的编码形式。

c)用 SOS 算法确定个体在解空间中的位置 x_i，更新差异度 θ。

d)学习操作：

（a）趋向学习。由式(16)~(18)产生趋向个体，若新个体适应度优于当前个体，则以新个体代替当前个体。

（b）交叉学习。由式(19)产生交叉学习个体，若新个体适应度大于当前个体，则以新个体代替当前个体。

e)变异操作。由式(20)~(21)计算变异信息，并产生变异个体，随机替换种群中 n_2 个体。

f)判断是否达到终止条件，达到则进入 g)，否则返回 c)。

g)算法终止，记录最优数据。

由上述步骤可知，RQSOS 是一种具有反馈更新机制的算法，如图 6 所示。

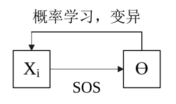


图 6　RQSOS 图示

染色体中，SOS 给出的基因取值具有非确定性，由量子位概率决定具体取值方式和大小。故 RQSOS 模式中染色体能以一种叠加态来描述解空间，能够保持解的多样性。

3　实验与分析

本章以数值优化问题和云工作流任务调度问题为例，与其他算法进行对比分析，验证本文算法性能。

3.1　数值实验及分析

实验采用表 1 所列 12 个基本测试函数验证算法性能，表中 X_i 为取值范围，B 为最优值，D 为函数所取维数，*Threshold* 为求解精度。

实验对 QPSO、混合粒子群算法(PSO-GA)、实数编码量子差分演化算法(RQDE)和 SOS 的优化性能和结果进行比对，本文算法参数 $q=12,a=0.25,n_1=5,n_2=3,\eta=P_l=P_c=rand,\Delta\theta_0=0.04\pi$，各算法独立运行 30 次，最大迭代次数为 10000，当算法达到规定精度或最大迭代次数时终止，记录 30 次实验平均值与方差并依据实验结果记录算法运行成功次数。表 2 列出了五种算法的实验结果，其中 $w/t/l$ 表示本文算法胜过、持平和劣于对比算法的函数个数，M 为均值，V 代表方差。

由表 2 可知，五种算法中本文算法表现最优，获得了 7 个测试函数的全局最优解，11 个测试函数最优解，而基本 SOS 算法只获得了 3 个测试函数的全局最优解，6 个测试函数最优解，并且在 12 个测试函数上 RQSOS 性能均优于等于基本 SOS 算

法，除在优化 f6 函数时本文得到优化效果微差于 QPSO 外，均优于其他算法，证明本文规则是有效的。

表 3 展示了五种算法在不同测试函数上的执行成功率，可以看出 RQDE、SOS 和本文算法 RQSOS 在执行成功率上优于 PSO-GA 和 QPSO 算法，且 RQSOS 算法成功率最高。说明 SOS 算法的探索能力强、鲁棒性好的优点，也进一步说明了本文提出算法保持了 SOS 算法优点，克服了算法容易陷入早熟的缺点，能达到更好的优化效果。

图 7 和 8 展示了五种算法在优化函数 f_2 和 f_8 的收敛曲线对比图。由图可知，在不同函数优化的问题上，不同算法的展现了不同的性能，本文算法均有较高的收敛能力和较强的探索能力。RQSOS 能产生优良解且具有较好收敛能力的原因为采用了学习模式和变异模式，学习模式加快种群向最优解方向发展，而变异模式保持了种群的多样性，故本文算法能够均衡算法的勘测能力和收敛能力，加强算法性能。

表 1　数值实验所用测试函数相关信息

funs	name	D	X	B	threshold
f_1	Sphere	30	[-100,100]	0	1.00e-006
f_2	Rosenbrock	30	[-30,30]	0	1.00e-001
f_3	Ackley	30	[-32,32]	0	1.00e-006
f_4	Griewank	30	[-600,600]	0	1.00e-006
f_5	Rastrigrin	30	[-5.12,5.12]	0	1.00e-006
f_6	Dixon-price	30	[-10,10]	0	1.00e-006
f_7	Zakharov	10	[-5,10]	0	1.00e-006
f_8	Michalewicz_{10}	10	[0,pi]	-9.6602	-
f_9	Peaks	2	[-3,3]	8.1062	-
f_{10}	Schaffers f_6	2	[-5,5]	0	1.00e-006
f_{11}	Schaffers f_7	2	[-100,100]	0	1.00e-006
f_{12}	Shubert	2	[-10,10]	-186.7309	-

表 2　本文算法与其他四种算法的仿真结果比较

Funs.	PSO-GA M±V	QPSO M±V	RQDE M±V	SOS M±V	RQSOS M±V
f_1	2.83e-00±4.01e-00	6.83e-25±1.98e-48	1.28e-18±2.16e-36	**3.01e-134±3.63e-267**	**9.20e-169±1.02e-280**
f_2	3.23e+02±1.13e+04	2.49e+01±1.54e-01	7.69e-00±1.39e+01	2.60e+01±1.41e-00	**1.71e-3±6.52e-06**
f_3	1.99e+01±0.00e-00	1.79e-14±1.21e-29	3.64e-10±1.69e-20	**8.89e-16±3.89e-62**	**8.89e-16±3.9e-62**
f_4	5.76e-01±8.78e-02	3.12e-02±5.28e-04	1.09e-12±2.91e-26	**0.00±0.00e-00**	**0.00±0.00e-00**
f_5	9.95e-02±8.91e-02	2.21e+01±4.57e+01	9.95e-02±8.91e-02	9.95e-02±8.91e-02	**0.00±0.00e-00**
f_6	3.96e-00±1.61e-00	**0±0.00e-00**	8.10e-03±5.90e-04	3.96e-00±1.61e-00	1.09e-05±4.70e-06
f_7	1.20e+01±1.31e+03	**3.33e-61±9.45e-121**	**3.98e-66±1.19e-131**	**4.27e-290±0.00e-00**	**0.00±0.00e-00**
f_8	-9.6171±1.20e-03	-6.8226±7.88e+01	-9.6396±3.87e-04	-9.6248±9.62e-04	**-9.6600±1.45e-07**
f_9	**8.1062±3.16e-30**	5.5428±2.83e-00	**8.1062±3.15e-30**	**8.1062±3.16e-30**	**8.1062±0.00e-00**
f_{10}	3.01e-02±0.00e-00	**0±0.00e-00**	**0.00±0.00e-00**	**0.00±0.00e-00**	**0.00±0.00e-00**
f_{11}	**0.00±0.00e-00**	**0±0.00e-00**	**0.00±0.00e-00**	**0.00±0.00e-00**	**0.00±0.00e-00**
f_{12}	-186.7308±1.04e-08	-186.7283±5.84e-005	**-186.7309±0.00e-00**	**-186.7309±0.00e-00**	**-186.7309±0.00e-00**
W/t/l	11/1/0	9/2/1	8/3/0	7/5/0	-

表 3　五种算法的执行成功率对比

Funs.	PSO_GA	QPSO	RQDE	SOS	RQSOS
	SR%				
f_1	17	100	100	100	100
f_2	0	0	30	10	87
f_3	0	100	100	100	100
f_4	37	23	100	100	100
f_5	23	17	87	100	100
f_6	0	100	90	17	100
f_7	93	100	100	100	100
f_8	30	7	60	23	83
f_9	100	0	100	100	100
f_{10}	90	100	100	100	100
f_{11}	67	100	100	100	100
f_{12}	83	50	100	100	100

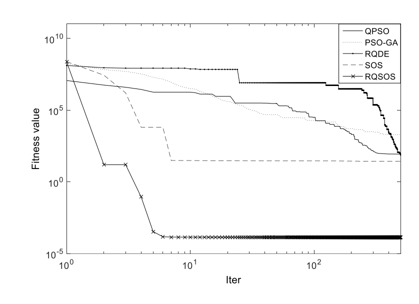


图 7　Rosenbrock 收敛曲线对比图

图 9 和 10 为 SOS 和 RQSOS 算法在优化函数 f_2 和 f_8 时种群差异度的对比，可知 SOS 算法优化 f_2 函数时种群多样性很快丢失，探寻最优解的能力下降，故其优化精度和执行成功率均很低；在优化 f_8 时，虽多样性可保持在一定范围，但其向最优靠近的速度很慢，因此，在规定迭代次数内很可能不能搜寻到最优解。很明显，本文策略在保持高收敛性的同时能够将种群

多样性控制在一定范围，进一步证明了算法的有效性。

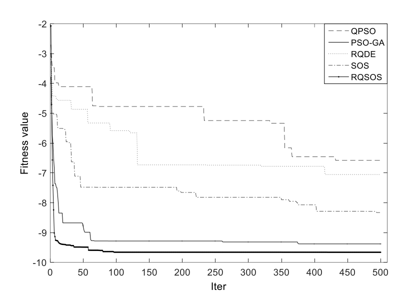


图 8　Michalewicz10 收敛曲线对比图

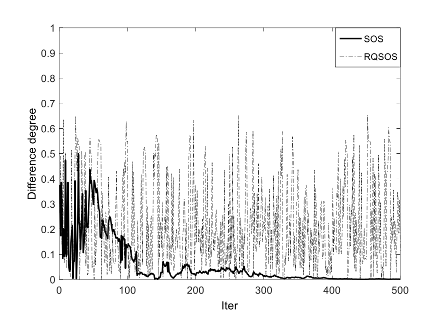


图 9　处理 f_2 函数 SOS 与 RQSOS 算法 θ 值对比

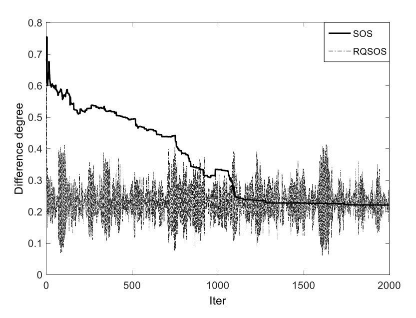


图 10　处理 f_8 函数 SOS 与 RQSOS 算法 θ 值对比

3.2　本文算法在云工作流任务调度中的应用

目前云环境中工作流调度算法从不同角度对工作流进行处理，从而缩短进程的总完成时间，提升系统性能。工作流任务可由 DAG(DAG={N,S})图描述，其中 N 由 n 个节点集合构成，n 为任务数，S 由 s 个边的集合构成，代表任务间的约束条件。n_i,n_j 间的 $S_{ij}\in S$ 表示任务 n_j 必须在任务 n_i 完成之后才能执行。本文将处理成本描述为任务的执行时间和使用处理器的费用。

图 11 为一个简单 DAG 图，非负权重 $C_{i,j}$ 关联 $S_{ij}\in S$ 边对应 n_i,n_j 间的通信量。如果任务依赖于不同的处理器，则需计算处理器间的通信花费，处于相同处理器间任务的通信花费认为是 0。

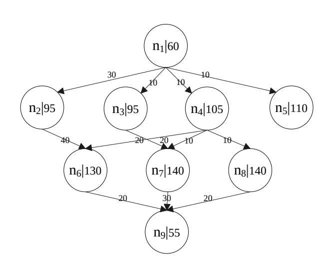


图 11　一个简单的 DAG 图

处于处理器 P_j 上的任务 n_i 的最早的开始时间 $T_{start}(n_i,p_j)$定义为

$$T_{start}(n_i,p_j)=\begin{cases}0, & if\ n_i为入口任务;\\ \max\{T_{free}(p_j),T_{ready}(n_i,p_j)\}, & else;\end{cases} \tag{22}$$

则任务 n_i 的完成时间为

$$FT_{p_j}(n_i)=T_{start}(n_i,p_j)+W(n_i)/W(p_j) \tag{23}$$

其中：$W(n_i)$为任务 n_i 计算量，$W(p_j)$代表处理器的处理能力。任务 n_i 为任务 n_j 的前继任务，p_x,p_y 分别为任务 n_i,n_j 所在处理单元，若 $i=j$ 则 $C(n_i,n_j)=0$，故 n_i 到达 n_j 的时间为

$$AT_{p_y}(n_i,n_j)=FT_{p_x}(n_i)+C(n_i,n_j)/C(p_x,p_y) \tag{24}$$

$$EST_{p_y}(n_j)=\max\{\max\{AT_{p_y}(n_i,n_j)\},FT_{p_y}(n_q)\}\} \tag{25}$$

其中：n_q 为处理器上最后完成的任务，在相同处理器上任务执行的优先级关系由式(26)确定。

$$\Pr i(n_i)=W(n_i)/M_{p\ell}+\max_{n_j\in succ(n_i)}\left(C(n_i,n_j)/M_c+\Pr i(n_j)\right)+ \alpha\cdot BL(n_i) \tag{26}$$

其中：M_p,M_c 为处理器处理能力中值和传输能力中值 $succ(n_i)$代表任务 n_i 的后继点集合，$BL(n_i)$为任务 n_i 的 Bottom-Level。

故本文的适应度函数可以由公式(27)-(29)表示，其中 $\alpha,\beta\in[0,1]$, $\alpha+\beta=1$，ε 为调整系数，T_{pj} 为占用第 j 个处理器的总时间，P_{pj} 为第 j 个处理器单位时间的使用价格。可根据实际应用而调整时间和花费所占权重。

$$Fitness=\alpha\cdot Time+\beta\cdot\varepsilon\cdot SP \tag{27}$$

$$Time=\max_{i=1}^{n}\left(EST_p(n_i)\right) \tag{28}$$

$$SP=\sum_{j=1}^{m}T_{p_j}\cdot P_{p_j} \tag{29}$$

为验证本文算法在处理云工作流调度问题时的有效性，对处理器和任务数变化时的调度问题进行模拟仿真，并分别与 QGA 和改进量子粒子群算法(Improved Quantum Particle Swarm Optimization, IQPSO)进行对比。实验中，算法最大迭代次数为 400 或 600 次，QGA 和 IQPSO 种群规模为 60，本文算法种群规模为 30，趋向个体数目为 3，变异个体数目为 5，圈数 q 为 7，其他参数与上节相同。

本次实验采用实验室设计的云任务调度仿真系统，依据用户制定参数，随机生成 10-100 个节点的任务和 4-10 个资源的系统。节点计算量控制在[50,200]内，节点间通信量控制在[5,30]内，节点最大入度和出度为 10，图最大深度为 8；资源计算能力取值范围[30,100],资源单位时间使用价格范围[1,10]，传输能力取值范围[10,30]。

如图 12 和图 13 为当资源数固定为 8 时的仿真结果，由图可以看出 IQPSO 算法寻得的完工时间效果比 QGA 差，但使用处理器价格要优于 QGA，而本文算法在时间和花费上均优于

IQPSO 和 QGA 算法。调度算法在约束条件不同的情况下，可能产生求得的完工时间与花费不均衡的现象，而本文算法具有更好的均衡性。

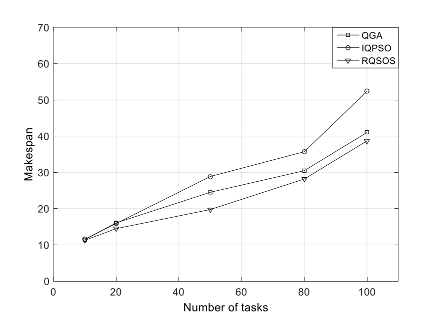


图 12　不同任务数时完工时间对比

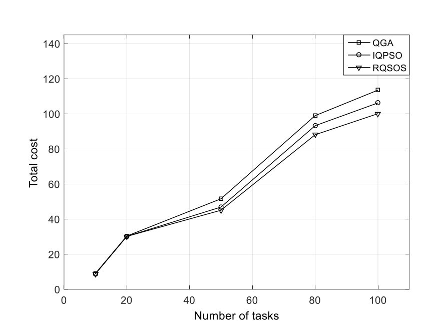


图 13　不同任务数时最小花费对比

图 14 和 15 展示了当任务数固定时，完工时间和最小花费随资源数变化而改变的结果，可看出在资源数少于 6 时本文算法得到的完工时间多于 QGA 和 IQPSO 算法，当资源数超过 6 后本文算法所用时间较少；对于最低花费，本文算法所得结果均优于 QGA 和 QIPSO 算法，故在任务数固定资源数改变时，本文算法依然具有较优性能。

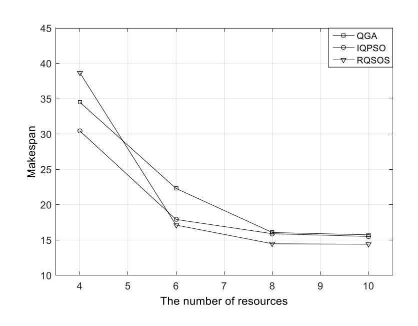


图 14　不同资源数时的完工时间对比

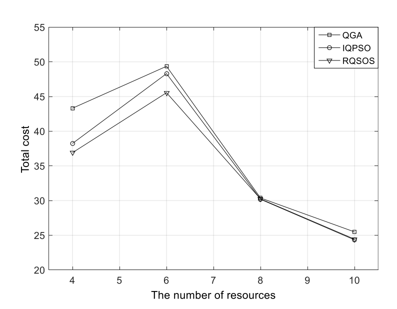


图 15　不同资源数时的最小花费对比

图 16 和 17 展示了任务数，资源数固定时三种算法适应度值变化的情况，由结果可知，IQPSO 算法和 RQSOS 算法在前期进化效率较高，但 IQPSO 算法性能很快下降，RQSOS 算法保持较高进化效率且优于 IQPSO 算法，说明本文算法所提出的学习模式和变异模式是有效的。

在任务数增加时，任务处理变得复杂，算法想寻得一个稳定点均需要一定迭代次数，但由图 17 可看出，本文算法能够很快的找到较优解，在算法后期，IQPSO 和 QGA 算法均有跳出局部最优的能力，但是效率极低，本文算法在后期以更大概率跳出局部最优，找到更优解，进一步说明了本文算法的优越性能。

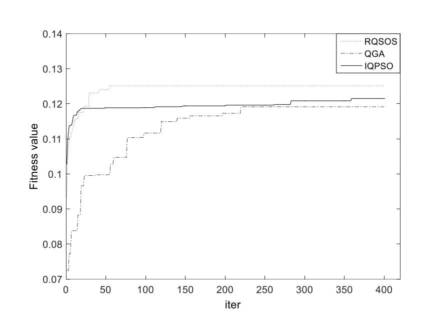


图 16　任务数为 20 时适应度和迭代次数对比

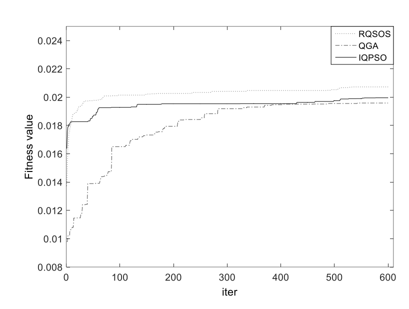


图 17　任务数为 80 时适应度和迭代次数对比

4　结束语

结合量子遗传算法理论与共生演算法，提出一种实数编码的量子共生演算法。首先提出差异度概念和基于阿基米德线的探索学习模式，可使算法在解空间快速、细致的搜索最优解。通过监测差异度值实时了解种群的多样性和与当前最优均值的距离，当种群多样性较强但综合距离当前最优均值距离较远时，采用趋向学习和交叉学习模式加快种群向最优方向的进化；当种群多样性丢失，为防止算法陷入局部最优，引入变异操作，加强了算法的探测能力。数值实验和云工作流任务调度实验结果表明，本文算法有效的提升了 SOS 算法性能，能够得到更优解。但监测信息是以反馈方式作用于算法，该方式增加了原算法复杂度。因此，在以后的研究工作中有必要进一步分析模型，加强本文提出思想与原算法的融合程度，减小算法的复杂度。

参考文献：

[1]　Narayanan A, Moore M. Quantum-inspired genetic algorithm [C]// Proc of

IEEE International Conference on Evolutionary Computation. Piscataway: IEEE Press, 1996: 61-66.

[2] Han K H, Jong H K. Quantum-inspired evolutionary algorithm for a class of combination optimization [J]. IEEE Trans on Evolutionary Computation, 2002, 6 (6): 580-593.

[3] Yang Shuyuan, Jiao Licheng. The quantum evolutionary programming [C]// Proc of the 5th International Conference on Computational Intelligence and Multimedia Applications. 2003: 362-367.

[4] Sun Jun, Xu Wenbo, Feng Bin. A global search strategy of quantum behaved particle swarm optimization [C]// Proc of IEEE Conference on Cybernetics and Intelligent Systems. 2004: 325-331.

[5] 李盼池，李世勇. 求解连续问题空间优化问题的量子蚁群算法 [J]. 控制理论与应用, 2008, 25 (2): 237-240.

[6] Luciano R S, Ricardo T, Marley M V. Quantum inspired evolutionary algorithm for ordering problems. Expert Systems with Applications, 2017, 67: 71-83.

[7] Pavithr R, Gursaran S. Quantum inspired social evolution (QSE) algorithm for 0-1 knapsack problem [J]. Swarm and Evolutionary Computation, 2016, 29: 33-46.

[8] Liu Min, Zhang Feng, Ma Yunlong, et al. Evacuation path optimization based on quantum ant colony algorithm [J]. Advanced Engineering Informatics, 2016, 30 (3): 259-267.

[9] Yuan Xiaohui, Wang Pengtao, Yuan Yanbin, et al. A new quantum inspired chaotic artificial bee colony algorithm for optimal power flow problem [J]. Energy Conversion and Management, 2015, 100: 1-9.

[10] Konar D, Bhattacharyya S, Sharma K, et al. An improved quantum-inspired genetic algorithm (HQIGA) for scheduling of real-time task in multiprocessor system. Applied Soft Computing, 2017, 53: 296-307.

[11] Zhao Shuanfeng, Xu Guanghua, Tao Tangfei, et al. Real-coded chaotic quantum inspired genetic algorithm for training of fuzzy neural networks. Computers and Mathematics with Applications, 2009, 57 (11-12): 2009-2015.

[12] 高辉，徐光辉，张锐，等. 实数编码量子进化算法 [J]. 控制与决策, 2008, 23 (1): 87-90.

[13] 高辉，张锐. 改进实数编码量子进化算法及其在参数估计中的应用 [J]. 控制与决策, 2011, 26 (3): 418-422.

[14] Cheng Minyuan, Doddy P. Symbiotic organisms search: a new metaheurstic optimization algorithm [J]. Computer and Structures, 2014, 139: 98-112.

[15] 赵克勤. 集对分析及其初步应用 [M]. 杭州：浙江科技出版社, 2000.

[16] Rahnamayan S, Wang G G, Ventrescaet M. An intuitive distance based explanation of opposition-based sampling [J]. Applied Soft Computing, 2012, 12 (9): 2828-2839.

[17] 刘会超，吴志健. 基于旋转学习机制的差分演化算法 [J]. 电子学报, 2015, 43 (10): 2040-2046.